AMENDMENTS IN THE CLAIMS

Please amend the claims as follows:

1. (currently amended) In a distributed processing system, a~~A~~ method for providing scalable device driver services within a control system of a network processor services architecture, said method comprising the steps of:

in response to determining a desired network processor functionality, loading one or more of a plurality of functional components, wherein said one or more functional components provide the desired network processor functionality and ~~that~~ wherein each of said plurality of functional components are independently selectable and provide a respective one of a plurality of network processor services;

providing at least one utility interposed between said one or more of said plurality of functional components and an operating system (OS) of said control system, wherein said at least one utility ~~that~~ provides an OS-independent communication interface for said plurality of functional components and enables bi-directional communication between said network processor functionality and said OS; and

in response to a receipt of a packet at said control system, ~~handling said packet utilizing one of said plurality of individual software components by~~ first routing said packet through said utility, wherein said packet is decoded into a common code understandable by said OS and said one of said plurality of ~~individual software~~ functional components, then handling said packet utilizing a selected one of said plurality of functional components, wherein said utility also converts all low and high level APIs of various components into a specific communication type utilized by the network processor functionality.

2. (original) The method of Claim 1, wherein said loading step includes the step of loading external application programming interfaces (APIs), low level APIs, and physical transport interfaces of a device driver.

3. (currently amended) The method of Claim 2, further comprising the steps of:

loading a customer definable service component within said external (APIs) that includes a customer desired network service, which is operable within said network processor services architecture; and

dynamically expanding available services to include said customer definable service component on-the-fly.

4. (original) The method of Claim 1, wherein said providing step includes the step of providing a bi-directional connection between said utility and said operating system, one or more network processors, and each of said functional components.

5. (original) The method of claim 1, wherein said providing step further comprises the step of linking a system services utility to said operating system, wherein, said system services utility operates to allow each of said individual software coded components to communicate with said OS.

6. (original) The method of Claim 1, wherein said providing step further provides a translation utility, which translates all incoming and outgoing service requests into a common network processor language to permit seamless connection and correspondence between said one or more network processors, said operating system, and each of said functional components to enable handling of network packets.

7. (currently amended) A computer program product for providing scalable device driver services within a control system of a network processor services architecture comprising:

a computer readable medium; and

program instructions on said computer readable medium for:

implementing a plurality of individual software components that each provide a respective one of a plurality of network processor services;

providing at least one utility interposed between said plurality of individual software components and an operating system (OS) of said control system, that provide an OS independent communication interface for said plurality of individual software components; and

in response to a receipt of a packet at said control system, ~~handling said packet utilizing one of said plurality of individual software components, by~~ first routing said packet through said utility, wherein said packet is translated into a code understandable by said OS and said one of said plurality of individual software components, then handling said packet utilizing one of said plurality of individual software components, wherein said utility also converts all low and high level APIs of various components into a specific communication type utilized by the network processor functionality.

8. (original) The computer program product of Claim 7, wherein program instructions for said loading step includes program instructions for loading external application programming interfaces (APIs), low level APIs, and physical transport interfaces of a device driver.

9. (currently amended) The computer program product of Claim 8, further comprising program instructions for:

loading a customer definable service component within said external APIs that includes a customer desired network service, which is operable within said network processor services architecture; and

dynamically expanding available services to include said customer definable service component on-the-fly.

10. (original) The computer program product of Claim 7, wherein said program instructions for said providing step includes instructions for providing a bi-directional connection between said utility and said operating system, one or more network processors, and each of said functional components.

11. (original) The computer program product of claim 7, wherein said program instructions for said providing step further comprises instructions for linking a system services utility to said operating system, wherein, said system services utility operates to allow each of said individual software coded components to communicate with said OS.

12. (original) The computer program product of Claim 7, wherein said program instructions for said providing step further includes program instructions for a translation utility, which translates all incoming and outgoing service requests into a common network processor language to permit seamless connection and correspondence between said one or more network processors, said operating system, and each of said functional components to enable handling of network packets.

13. (currently amended) A control system of a network processor services architecture comprising:

a plurality of individually loadable functional components within a device driver of said control system that each represents a network processor service;

at least one utility for enabling each of said functional components to communicate with an operating system (OS) of said control system, wherein ~~said utility~~, in response ~~responsive~~ to a receipt of a request by said OS to perform a network processor function, said utility translates said request into a call of a particular one of said plurality of functional components that administers said network processor services of one or more network processors, wherein said utility is located within lower level APIs and is communicatively coupled to external APIs, OS, software stack, and a network processor resource services, and wherein said utility also converts all low and high level APIs of various components into a specific communication type utilized by the network processor functionality; and

processing hardware that executes code of said OS, said utility and said functional components.

14. (original) The system of Claim 13, wherein said plurality of functional components includes external application programming interfaces (APIs), low level APIs, and physical transport interfaces of a device driver.

15. (currently amended) The system of Claim 14, further comprising a customer definable service component within said external (APIs) that includes a customer desired network service, which is operable within said network processor services architecture, wherein available services are dynamically expandable to include said customer definable service component on-the-fly.

16. (original) The system of Claim 13, wherein said utility includes a system services utility coupled to said operating system that operates to allow each of said individual software coded components to communicate with said OS.

17. (original) The system of Claim 13, wherein said utility includes a translation utility, which translates all incoming and outgoing service requests into a common network processor language to permit seamless connection and correspondence between said one or more network processors, said operating system, and each of said functional components to enable handling of network packets.

18. (original) A network processing services architecture comprising:

one or more network processors; and

a control system, coupled to said one or more network processors, having a scalable device driver for controlling interactions with said one or more network processors, wherein said scalable device driver is comprised of a plurality of individual functional software components including a plurality of functional elements for network processing and a conversion utility to enable compatibility between said plurality of functional elements and said one or more network processors to effectively communicate and interact within any processor type and OS to effectuate the transfer of frames and control information between said control system and said one or more network processors.

19. (original) The network processing services architecture of Claim 18, further comprising a switch fabric coupled to said one or more network processors.

20. (original) A device driver program for a distributed processing system comprising:

a plurality of system services programs providing high level bi-directional communication between a common code program and each of a network transport manager, a network resource manager and a control message formatter, said network transport manager providing low level bi-directional communication to a physical transport processor, said physical transport processor providing primitive bi-directional communication to one or more network processors.